ARTICLE OPEN

Check for updates

A quantum algorithm for string matching

Pradeep Niroula[1,2]✉ and Yunseong Nam[3]✉

Algorithms that search for a pattern within a larger data-set appear ubiquitously in text and image processing. Here, we present an explicit, circuit-level implementation of a quantum pattern-matching algorithm that matches a search string (pattern) of length M inside a longer text of length N. Our algorithm has a time complexity of $\tilde{O}(\sqrt{N})$, while the space complexity remains modest at $O(N+M)$. We report the quantum gate counts relevant for both pre-fault-tolerant and fault-tolerant regimes.

npj Quantum Information (2021)7:37 ; https://doi.org/10.1038/s41534-021-00369-3

INTRODUCTION

Pattern matching is one of the core algorithms in computer science that stand to benefit from quantum computers[1,2]. Pattern matching algorithms are used ubiquitously used in image processing[3,4], the study of DNA sequences[5], and data compression and statistics[6], to name a few. Thus, accelerating pattern matching using a quantum computer would be a boon to all these areas.

The simplest form of pattern matching is string matching. In string matching, given a long string $\mathcal{T}$ of length N, we search for a pattern $\mathcal{P}$ of length M with $M \le N$[7]. Depending on the application, we may need to search for an exact match or a fuzzy match, or a match with some wildcards[8].

The best known classical algorithm for string matching is the Knuth-Pratt-Morris algorithm, which has the worst-case time complexity of $\Theta(N+M)$[9,10]. The best-known algorithms for approximate string matching have a similar run-time of $\Theta(N+M)$. For random strings, the exact matching complexity is lower bounded by $\Omega((N/M)\log(\mathrm{M}))$[11].

Ramesh and Vinay developed an exact string matching quantum algorithm with a query complexity of $\tilde{O}(\sqrt{N}+\sqrt{M})$[1]. This algorithm uses Grover's search to identify the position at which a segment of length M from $\mathcal{T}$ matches the pattern $\mathcal{P}$, where each of the checks is done using a nested Grover search. However, this work does not construct explicit oracles required and the total time complexity, measured in units of gate depth, is bound to increase once we account for the gate-level complexity of accessing the text and pattern from a database. Another approach that relies on a quantum solver for the dihedral hidden subgroup problem[12] has a time complexity of $\tilde{O}((N/M)^{1/2}2^{O(\sqrt{\log(M)})})$ for average-case matching[13]. This work also assumes that M is larger than the logarithm of the length N, i.e $M=\omega(\log \mathrm{N})$ and fails with a high probability for certain worst-case inputs. In our work, we do not make any assumptions on the length of pattern or the distribution of inputs.

In this paper, we present a string-matching algorithm, based on generalized Grover's amplitude amplification[14], with a time complexity of $\tilde{O}(\sqrt{N})$ for arbitrary text length N and pattern length $M \le N$. Note our algorithm does not rely on a quantum database, incurring no initialization overhead of the database, expected to be $O(N)$, that would overshadow any quantum advantage. The techniques we develop for our algorithm can readily be extended to solve pattern matching problems in higher dimensions. Over the course of detailing each step of our algorithm, we also ensure to provide a gate–by–gate level instruction to construct relevant quantum circuits. This allows us to straightforwardly obtain a concrete estimate of the total gate counts. The gate counts we report help us establish contexts as to when we may expect quantum computers to be of help in the problem space of pattern matching.

Our paper is organized as follows. To motivate the readers, we first compare our main results that are derived in the remainder of the paper with the current state of the art. After the comparison, we provide an outline of our string-matching algorithm. In Section "Results", we provide the details of the algorithm, including the explicit circuits for all necessary oracles. We then calculate the overall complexity of our algorithm. We provide an estimate for gate counts in terms of CNOT and T gates, useful for pre-fault tolerant and fault tolerant regimes, respectively. We summarize our paper in Section "Discussion" and discuss the implications of our results.

We start by pointing out that our work differs from[13], where the algorithm therein targets an average case input, in that we, as in[1], provide a quantum algorithm for pattern matching for the worst case inputs. The work in[13] further assumes $M=\omega(\log(\mathrm{N}))$, whereas the work in[1] and the work reported in this manuscript do not. We rely on a Grover oracle (see Section "Grover oracle") that simply checks if a state is an all-zero state in the computational basis, whereas the oracles in refs [1,13] are random memory access oracles of the form $\sum_i|i\rangle|0\rangle \rightarrow \sum_i|i\rangle|t_i\rangle$ where t_i is the ith bit of a text. As such, we are unaware of an efficient quantum circuit that implements the oracle (see Section G.4 of the appendix of ref. [15] for the best-known construction) without resorting to quantum random access memory (QRAM)[16]. The known blueprints for QRAM[16] have polylogarithmic time complexity in the size of memory to be accessed. In our case, the size of memory is $O(N)$ and, therefore, QRAM queries will incur at additional multiplicative cost of at least $O((\log \mathrm{N})^2)$. Moreover, we would also have to account for the cost of initializing the quantum memory—this is expected to take a number of operations linear in N[17]. In contrast, our algorithm does not assume any random access oracles. We also provide an explicit circuit for the Grover oracle we need using elementary quantum gates, specifically single-qubit Clifford, T, and CNOT gates.

Note the algorithm in ref. [13] fails with a probability $O(1/N)$ over the choice of $\mathcal{T}$ and $\mathcal{P}$. For certain worst-case $\mathcal{T}$ and $\mathcal{P}$, the algorithm inherently fails to return a match. In addition, there is internal randomness in the algorithm which contributes to an

[1]Joint Quantum Institute, NIST/University of Maryland, College Park, MD, USA. [2]Joint Center for Quantum Information and Computer Science, NIST/University of Maryland, College Park, MD, USA. [3]IonQ, College Park, MD, USA. ✉email: pniroula@umd.edu; nam@ionq.co

Table 1. Comparison of our work with prior algorithms discussed in this paper.

Paper	Query complexity	Oracle details	Time complexity	Comment
[1]	$O(\sqrt{N}\log(\sqrt{N/M})\log M+\sqrt{M}(\log M)^2)$	Random access oracle	–	Worst case
[13]	$O((\sqrt{N/M})2^{(3/2)\sqrt{(2\log_2 3)\log_2 M}}(\log M)^{3/2}\log N)$	Random access oracle	–	Average case
This work	$O(\sqrt{N})$	Explicit oracle provided	$O(\sqrt{N}((\log N)^2+\log(M)))$	Worst case

The oracles for refs [13] and [1] provide random access to bits in the text and pattern. This random-access oracle is not needed in our work. Instead, for our work, by oracle, we mean a Grover oracle that checks if a register is in an all-zero state. We provide an explicit construction for such an oracle. The time complexities for refs [1] and [13] are unknown because the time of execution depends on the random-access oracles, which do not have a circuit-level construction in the respective papers.

additional probability of failure. Our work also fails with probability $O(1/N)$ if there is a match between $\mathcal{T}$ and $\mathcal{P}$, but this is purely due to the internal randomness of Grover's algorithm. We can simply repeat the algorithm to suppress the failure probability to be arbitrarily small, with the average repetition number of $N/(N-1)$. We make no assumptions on the distribution of text and pattern and the algorithm works for all possible inputs. This may be contrasted to the impossibility to suppress the failure probability by repeated use of the algorithm for the worst-case inputs in ref. [13].

Our algorithm has a space complexity of $O(N+M)$ since we need N (M) qubits to store the text (pattern). With $N>M$, we may omit the M dependence and simplify it to $O(N)$. The space complexities of[1,13] depend on the space complexity of the oracle. Assuming an N-bit register containing the text to be searched over is prepared in QRAM, in the bucket-brigade model, the bulk of the space complexity comes from routing qutrits, where random access over N bits of information requires $O(N)$ routing qutrits. Expending a constant number of qubits for each qutrit, the space complexities of[1,13] are $\Omega(N)$, and likely $\Theta(N)$.

Finally, unlike the two prior works, the simplicity of our algorithm allows us to not just provide an explicit circuit-level blueprint for the algorithm but also estimate the quantum resources needed to implement it. A summary of the comparison between our work and[1,13] is given in Table 1.

In the remainder of this section, we outline the steps of our algorithm. The detailed implementation is presented in Section "Results".

1. Initialize two quantum registers to

   $$|t_0t_1t_2\ldots t_{N-1}\rangle|p_0p_1\ldots p_{M-1}\rangle,$$

   where t_i and p_i denote the ith bit of string $\mathcal{T}$ and pattern $\mathcal{P}$, respectively.
2. Transform the first register containing the string $\mathcal{T}$ into a superposition of N states, where each state is a bit-shifted state of the original state of the first register, shifted by 0, 1, 2..., $N-1$ bits. This results in, assuming modulo-N space for the bit indices,

   $$\left(\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}|t_{0+k}t_{1+k}t_{2+k}\ldots t_{N-1+k}\rangle\right)|p_0p_1\ldots p_{M-1}\rangle \tag{1}$$

3. Compute XOR between the first M bits of the first register and all M bits of the second register to obtain

   $$\begin{aligned}&\frac{1}{\sqrt{N}}\sum_k|t_{0+k}t_{1+k}\ldots t_{N-1+k}\rangle\\ &|(p_0\oplus t_{0+k})(p_1\oplus t_{1+k})\ldots(p_{M-1}\oplus t_{M-1+k})\rangle.\end{aligned} \tag{2}$$

4. The second register is all zeros if the pattern matches with the first M bits of $\mathcal{T}$. The register contains d ones if the string and the pattern differ in d bit positions.
5. Use the generalized Grover search or amplitude amplification[14] to isolate the state where the second register has all zeros (when searching for exact match) or has fewer than D matches (in the case of fuzzy search).

RESULTS

In this section, we lay out the detailed implementation of the algorithm we outlined above. Specifically, we detail the transformations and registers used to implement the algorithm. One of the central transformations to be used in our algorithm is the cyclic shift operator. We present the details of its construction in Section "Construction of the cyclic-shift operator." We also present the construction of the necessary Grover oracle in Section "Grover oracle" for completeness.

To encode a binary string $\mathcal{T}$ of length N and a binary pattern $\mathcal{P}$ of length M, we use quantum registers of N and M qubits, respectively. This can be done by using identity and bit-flip gates on a quantum register initialized as $|0\rangle^{\otimes(N+M)}$. Denoting the encoded states as

$$\begin{aligned}|\mathcal{T}\rangle&=|t_0t_1\ldots t_{N-1}\rangle=\bigotimes_{i=0}^{N-1}|t_i\rangle,\\ |\mathcal{P}\rangle&=|p_0p_1\ldots p_{M-1}\rangle=\bigotimes_{j=0}^{M-1}|p_j\rangle,\end{aligned} \tag{3}$$

where t_i (p_i) is the ith bit of string $\mathcal{T}$ ($\mathcal{P}$), together with an index register of n qubits in the zero states, we prepare on a quantum computer a composite initial state

$$|\psi\rangle=|0\rangle^{\otimes n}\left[\bigotimes_{i=0}^{N-1}|t_i\rangle\right]\left[\bigotimes_{j=0}^{M-1}|p_j\rangle\right], \tag{4}$$

where, for convenience, we assumed $N=2^n$. Next, we apply an n-qubit Hadamard transform $H^{\otimes n}$ (or a Fourier transform in case of $N\neq 2^n$ for $n\in\mathbb{N}$) on the index register to produce a uniform superposition of $|0\rangle,|1\rangle,\ldots|N-1\rangle$, i.e.,

$$\left(H^{\otimes n}|0\rangle^{\otimes n}\right)\left[\bigotimes_{i=0}^{N-1}|t_i\rangle\right]\left[\bigotimes_{j=0}^{M-1}|p_j\rangle\right]=\left(\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}|k\rangle\right)\left[\bigotimes_{i=0}^{N-1}|t_i\rangle\right]\left[\bigotimes_{j=0}^{M-1}|p_j\rangle\right]. \tag{5}$$

We now apply a cyclic shift operator $\mathcal{S}$ that left-circular shifts the qubits of the target state by k positions, where the values of k are encoded in the control state (see Section "Construction of the cyclic-shift operator" for details). Applying $\mathcal{S}$ on the first two registers results in

$$\begin{aligned}&\left[\mathcal{S}\left(\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}|k\rangle\right)\left(\bigotimes_{i=0}^{N-1}|t_i\rangle\right)\right]\left(\bigotimes_{j=0}^{M-1}|p_j\rangle\right)\\ &=\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}|k\rangle\left(\bigotimes_{i=0}^{N-1}|t_{i+k}\rangle\right)\left(\bigotimes_{j=0}^{M-1}|p_j\rangle\right).\end{aligned} \tag{6}$$

At this point, we check for the match between the cyclically-shifted text strings in the second register and the pattern string stored in the third register. We use an XOR operation between each of the first M bits of the second register with each of the M bits of the third register. For instance, if the XOR results are all zeros, the strings match. With the help of CNOT gates on a quantum computer then, we obtain, with an abuse of notation,

$$\begin{aligned}&\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}|k\rangle\mathrm{CNOT}^{\otimes M}\left[\left(\bigotimes_{i=0}^{N-1}|t_{i+k}\rangle\right)\left(\bigotimes_{j=0}^{M-1}|p_j\rangle\right)\right]\\ &=\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}\left[|k\rangle\left(\bigotimes_{i=0}^{N-1}|t_{i+k}\rangle\right)\left(\bigotimes_{j=0}^{M-1}|p_j\oplus t_{j+k}\rangle\right)\right].\end{aligned}\tag{7}$$

The final register, to this end, contains the number of mismatches between the pattern and the first M bits of the string register. Indeed, it is all zero if and only if those two string segments match completely.

We may now use the generalized Grover search or amplitude amplification[14] to search for the state where the pattern register is in $|0\rangle$ state (in the case of exact search). If this state is found, we know that the pattern occurs in the string. We also obtain the position from the index register where this match occurs. In addition to the exact match, we can also use this method to search for fuzzy matches or matches with wildcards by constructing appropriate Grover oracles.

Construction of the cyclic-shift operator

In this subsection, we explicitly construct a circuit that implements the cyclic-shift operator $\mathcal{S}$. The two-register operator $\mathcal{S}$ is defined according to

$$\mathcal{S}\left[|k\rangle\bigotimes_{i=0}^{N-1}|t_i\rangle\right]=\left[|k\rangle\bigotimes_{i=0}^{N-1}\mathcal{S}_k|t_i\rangle\right]=\left[|k\rangle\bigotimes_{i=0}^{N-1}|t_{i+k}\rangle\right].\tag{8}$$

To implement the k-controlled circular shift operator $\mathcal{S}_k$, we consider k in its binary encoded form $|k\rangle$ as $|k_0\rangle|k_1\rangle\ldots|k_{n-1}\rangle$, such that $2^0k_0+2^1k_1+\ldots+2^{n-1}k_{n-1}=k$. The circular bitwise rotation by k in the second register can then be implemented by a product of controlled-shift operators that shifts the target qubits by 2^j bits, conditioned on the k_jth qubit. Using $\mathcal{S}_a\mathcal{S}_b=\mathcal{S}_{a+b}$, we may now write

$$|k\rangle\bigotimes_{i=0}^{N-1}\mathcal{S}_k|t_i\rangle=\left(\bigotimes_{j=0}^{n-1}|k_j\rangle\right)\bigotimes_{i=0}^{N-1}\left(\prod_{j=0}^{n-1}\mathcal{S}_{2^j}^{(k_j)}\right)|t_i\rangle.\tag{9}$$

where $\mathcal{S}_{2^j}^{(k_j)}$ applies a shift of 2^j bits on the second register, which encodes the text $\mathcal{T}$, controlled by the jth qubit of the index register $|k\rangle$. The circuit decomposition of this as a visual guide is shown in Fig. 1.

The decomposition shown in (9) reveals that, together with (8), it suffices to now consider the controlled bit-shift operators $S_{2^j}^{(c)}$ that circular shifts by 2^j bits for some j conditioned on qubit c to implement the cyclic-shift operator $\mathcal{S}$. To this end, in order to construct the circuit for $S_{2^j}^{(c)}$, we first consider an operator S_{2^j} without any controls, which, as we show below, can be implemented using SWAP gates. We later promote the swap gates to a controlled version, effectively replacing the SWAP gates with controlled-SWAP (Fredkin) gates.

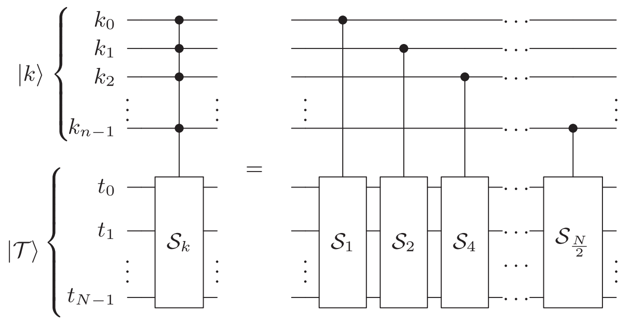

Fig. 1 Circuit diagram for circular bitwise rotation operator S_k. A shift by k bits can be achieved by a product of $\log(k)$ controlled shift operations.

A circular shift operator S_s by s bits applies a permutation P_s, in modulo N space, of the form

$$P_s=\{N-s,N-s+1,N-s+2,\ldots,N-s-1\},\tag{10}$$

where the $N-s$th bit is inserted in the zeroth position, $N-s+1$th bit is inserted in the first position, and so on. Any such permutation can be decomposed into a product of transpositions. As a result, a circular shift operation of the form (9) can be decomposed into a product of SWAP operations.

We now calculate how many SWAP-operation layers are needed to efficiently apply the permutation of the form (10). With a register with N qubits, we can apply $N/2$ SWAP operations in parallel. Using the $N/2$-parallel SWAP operator, we can move $N/2$ qubits to their right positions in a single time step. This leaves us with sorting the remainder of $N/2$ bits. At each subsequent time step, the number of qubits that need to be swapped decreases by half. Therefore, we can arbitrarily permute N qubits in $O(\log(\mathrm{N}))$ time steps using parallel SWAP operations. A sample diagrammatic representation of this unitary operation is shown in Fig. 2. This implies that each of the controlled shift operators $S_{2^j}^{k_j}$ in (9) can be achieved in $O(\log(\mathrm{N}))$ time steps using parallel controlled-SWAP operators.

We next discuss a method to apply as many as $N/2$ parallel swap operations, controlled on the same qubit in the index register. As shown below, we achieve this at the cost of $N/2$ clean ancilla qubits.

We start by considering a fan-out CNOT operation, acting on the control qubit in a state $|k_j\rangle$ and $N/2$ clean ancilla qubits initialized to $|0\rangle$ as targets. This results in $N/2$ copies of $|k_j\rangle$, which can then be used to implement up to $N/2$ Fredkin gates in a single time step. Once all necessary Fredkin gates have been implemented, we undo the fan-out operation and return all ancilla qubits to $|0\rangle$

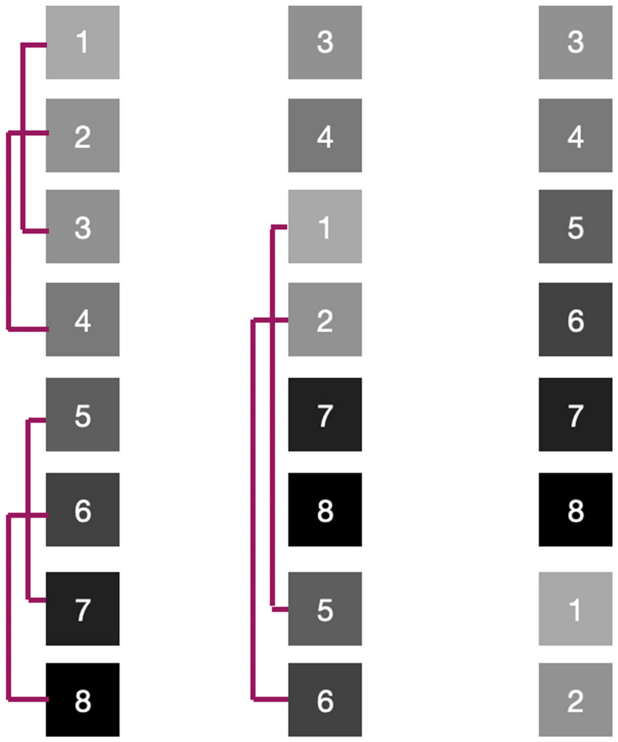

Fig. 2 A diagrammatic representation of the circular shift operator. In this example, we left circular shift a register of 8 qubits by 6 positions within two-time steps. This kind of operation can in general be performed in depth $\log(\mathrm{N})-1$ using parallel SWAP operations, where N is the size of the qubit register.

states. We recycle the freed-up ancilla qubits for the subsequent control qubits, one at a time.

The time cost of the fan-out operation is $O(\log(\mathrm{N}))$. Since there are $O(\log(\mathrm{N}))$ parallel SWAP layers required for the implementation of the qubit permutation discussed in Section "Construction of the cyclic-shift operator", the overall time complexity of $S_{2^j}^{(c)}$ is $O(\log(\mathrm{N}))$.

Grover oracle

To complete our algorithm, we need a Grover oracle U_w that acts on the pattern register, required to amplify and help identify exact matches or close matches. The oracle may be defined according to

$$U_w|x_0x_1\ldots x_{M-1}\rangle = \begin{cases} -|x_0x_1\ldots x_{M-1}\rangle & \sum_{i=0}^{M-1} x_i \le d, \\ +|x_0x_1\ldots x_{M-1}\rangle & \sum_{i=0}^{M-1} x_i > d, \end{cases} \tag{11}$$

where d is zero if we desire to find exact matches and a small number if we desire to find close matches. Assuming an architecture that has long-range interactions, we can obtain this oracle in $O(\log(M))$ depth using $O(M)$ ancilla qubits. We note in passing that there have also been proposals to implement a single-step n-control Toffoli that takes $O(1)$ time in trapped-ion and neutral-atom architectures[18]. For the remainder of the paper, however, we take the circuit-depth complexity of this oracle to be $O(\log(M))$.

Time complexity

In this subsection, we compute the time complexity of our algorithm. Encoding of strings $\mathcal{T}$ and $\mathcal{P}$ takes $O(1)$ time. The Hadamard transformation applied to the index register takes $O(1)$ time as well. The cyclic-shift operator $\mathcal{S}$ takes time $O((\log(N))^2)$, since each $S_{2^j}^{(k_j)}$ operator, including the fan-out and its uncompute operation, takes $O(\log(N))$ time and $j = 0, 1, 2, \ldots, \log(N) - 1$. The evaluation of XOR results via CNOT gates takes time $O(1)$, as it admits a straightforward parallel operation. Lastly, the Grover oracle has the complexity $O(\log(\mathrm{M}))$. The overall complexity of the steps considered so far, a single Grover step, is then $O((\log(N))^2 + \log(M))$.

For the Grover search to be successful, we need to repeat the Grover steps $O(\sqrt{N})$ times. This brings the total complexity to $O(\sqrt{N}((\log(N))^2 + \log(M)))$.

Space complexity

In addition to the N and M qubits needed to encode the search string and the pattern, we need $O(\log(\mathrm{N}))$ qubits for the index register. For the depth-optimized implementation of our algorithm we need $N/2$ ancilla qubits for the index register. Furthermore, $O(M)$ ancilla qubits are required for the depth-optimized Grover oracle implementation. Therefore, the space complexity of our string-matching algorithm is $O(N + M)$.

Gate counts

In this section, we obtain an estimate for the gate count in terms of CNOT and T gates. We chose the two gates as metrics since it is widely expected that two-qubit gates, such as CNOT, are expected to dominate the cost of implementation in the pre-fault tolerant regime, whereas T gates are expected to dominate the cost of implementation in the fault-tolerant regime, assuming the standard gate set of Clifford + T.

The strings $\mathcal{T}$ and $\mathcal{P}$ can be encoded in qubits initially in $|0\rangle$ state using only identity and bit-flip(X) gates and thus the encoding step has zero cost. A Hadamard transform of the index register in (5) needs $\log(\mathrm{N})$ Hadamard gates, requiring zero cost as well. The cyclic shift operator $\mathcal{S}$ in (6) consists of $\log(\mathrm{N})$ applications of $S_s^{(c)}$ operators. Each $S_s^{(c)}$ operator consists of a CNOT fan-out to $N/2 - 1$ target qubits, its inverse, and at most $N - 1$ Fredkin gates, since the permutation specified in (10) of size as large as N can be decomposed into at most $N - 1$ transpositions. As shown explicitly in Supplementary Note 1, based on circuit identities reported in refs [19,20], each Fredkin gate costs 7 CNOT gates and 7 T gates. Thus the cyclic shift operator costs at most $(8N - 9)\log(\mathrm{N})$ CNOT gates and $[7(N - 1)]\log(\mathrm{N})$ T gates. Next, the XOR operation in (7) takes M CNOT gates. Lastly, the Grover oracle of (11), using a parallelized version of the results reported in[21] (see Supplementary Note 2 for details), can be implemented with $6M - 12$ CNOT gates and $8M - 17$ T gates with a linear overhead in ancilla upper bounded by $M - 3$.

Finally, we need to repeat this $\sqrt{N}$ times for amplitude amplification. The total CNOT and T count is, thus, given by

$$\begin{aligned} \#\mathrm{CNOT} &= (7M - 12 + (8N - 9)\log(N)) \times 2\sqrt{N}, \\ \#\mathrm{T} &= (8M - 17 + 7(N - 1)\log(N)) \times 2\sqrt{N}, \end{aligned} \tag{12}$$

where the factor of 2 comes from the fact that for amplitude amplification, we need to apply a unitary to produce a state $|\psi\rangle = U|0\rangle$ and also the inverse unitary $U^\dagger$.

Based on (12), we see that searching for a pattern with 20 ASCII characters (or 160 bits) in a text file that is 1 MB long would require about 10^{13} CNOT and T gates. Similarly, searching for a kilobyte-long pattern of a genetic signature in a genome sequence of 1 GB would require more than 10^{17} CNOT and T gates. We expect classical computers to outperform quantum computers for datasets of such length. However, for applications like matching templates in data generated by gravitational-wave experiments which may be petabytes long (matching a megabyte-long signature in the petabyte-long text would require 10^{25} CNOT and T gates), we may expect to see the quantum advantage.

DISCUSSION

In this paper, we have constructed a quantum string-matching algorithm that admits a circuit-depth complexity of $O(\sqrt{N}((\log(\mathrm{N}))^2 + \log(\mathrm{M})))$. We also provide an explicit gate-level implementation of our algorithm, enabling a concrete estimate of quantum resources needed. The direct use cases of the matching algorithm range from a simple text search in a large file to detecting patterns in an image. The simple matching procedure can help, for example, in making intelligent recommendations based on pictures in a consumer device[22], detecting defects in industrial lithography[23], detecting signals in large time-series data collected in experiments like the Laser Interferometer Gravitational-Wave Observatory[24], etc. In these applications, the typical size of data to be searched varies between $\sim 10^6$ and $\sim 10^{15}$ bytes. Our algorithm admits processing of such data size in time steps $\sim \mathcal{C} \times (\log_2(N))^2\sqrt{N}$, where $\mathcal{C} < 20$ and N is the number of bits in the data. We hope the speed-up provided by the quantum algorithm contributes to further advances in these areas.

DATA AVAILABILITY

All data needed to evaluate the conclusions in the paper are present in the paper and/or the Supplementary Materials. Additional data related to this paper may be requested from the authors. Correspondence and requests for material should be addressed to Y.N. (nam@ionq.co).

Received: 10 June 2020; Accepted: 13 January 2021;
Published online: 16 February 2021

REFERENCES

1. Ramesh, H. & Vinay, V. String matching in $O(n + m)$ quantum time. *J. Discret. Algorithms* **1**, 103–110 (2003).
2. Sasaki, M., Carlini, A. & Jozsa, R. Quantum template matching. *Phys. Rev. A* **64**, 022317 (2001).

3. Landau, G. M. & Vishkin, U. Pattern matching in a digitized image. *Algorithmica* **12**, 375–408 (1994).
4. Bunke, H. & Bühler, U. Applications of approximate string matching to 2d shape recognition. *Pattern Recognit.* **26**, 1797–1812 (1993).
5. Chang, W. I. & Lawler, E. L. Sublinear approximate string matching and biological applications. *Algorithmica* **12**, 327–344 (1994).
6. Wyner, A. J. *String Matching Theorems and Applications to Data Compression and Statistics*. PhD Dissertation, (Stanford University, 1994).
7. Charras, C. & Lecroq, T. *Handbook of Exact String Matching Algorithms*. (Citeseer, 2004).
8. Singla, N. & Garg, D. String matching algorithms and their applicability in various applications. *Int. J. Soft Comput. Eng.* **1**, 218–222 (2012).
9. Knuth, D. E., Morris, J. H. Jr & Pratt, V. R. Fast pattern matching in strings. *SIAM J. Comput.* **6**, 323–350 (1977).
10. Hakak, S. I. et al. Exact string matching algorithms: Survey, issues, and future research directions. *IEEE Access* **7**, 69614–69637 (2019).
11. Yao, A. C. C. The complexity of pattern matching for a random string. *SIAM J. Comput.* **8**, 368–387 (1979).
12. Kuperberg, G. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM J. Comput.* **35**, 170–188 (2005).
13. Montanaro, A. Quantum pattern matching fast on average. *Algorithmica* **77**, 16–39 (2017).
14. Brassard, G., Hoyer, P., Mosca, M. & Tapp, A. Quantum amplitude amplification and estimation. *Quantum Computation and Information Vol. 305 of AMS Contemporary Mathematics Series* (eds Lomonaco, S. J. & Brandt, H. E.) 53–74, 2002.
15. Childs, A. M., Maslov, D., Nam, Y., Ross, N. J. & Su, Y. Toward the first quantum simulation with quantum speedup. *Proc. Natl. Acad. Sci. USA* **115**, 9456–9461 (2018).
16. Giovannetti, V., Lloyd, S. & Maccone, L. Quantum random access memory. *Phys. Rev. Lett.* **100**, 160501 (2008).
17. Park, D. K., Petruccione, F. & Rhee, J. K. K. Circuit-based quantum random access memory for classical data. *Sci. Rep.* **9**, 1–8 (2019).
18. Rasmussen, S. E., Groenland, K., Gerritsma, R., Schoutens, K. & Zinner, N. T. Single-step implementation of high-fidelity *n* -bit toffoli gates. *Phys. Rev. A* **101**, 022308 (2020).
19. Nam, Y., Ross, N. J., Su, Y., Childs, A. M. & Maslov, D. Automated optimization of large quantum circuits with continuous parameters. *npj Quantum Inf.* **4**, 1–12 (2018).
20. Nam, Y. et al. Ground-state energy estimation of the water molecule on a trapped-ion quantum computer. *npj Quantum Inf.* **6**, 33 (2020).
21. Maslov, D. Advantages of using relative-phase Toffoli gates with an application to multiple control Toffoli optimization. *Phys. Rev. A* **93**, 022311 (2016).
22. Yuan, C., Heller, G. S., Rybakov, O., Ramaswamy, S., & Thomas, J. O. *Object Recognition for Three-dimensional Bodies*, US Patent 9,424,461 (2016).
23. Chu, X., Lauber, J. A., & Runyon, J. R. *Detecting Defects on a Wafer Using Template Image Matching*, US Patent 9,311,698 (2016).
24. Owen, B. J. & Sathyaprakash, B. S. Matched filtering of gravitational waves from inspiraling compact binaries: computational cost and template placement. *Phys. Rev. D* **60**, 022002 (1999).

ACKNOWLEDGEMENTS

The authors would like to thank Jae Pak at IonQ for helpful conversations. P.N. acknowledges funding by the DoE ASCR Accelerated Research in Quantum Computing program (award No. DE-SC0020312) and the DoE ASCR Quantum Testbed Pathfinder program (award No. DE-SC0019040).

AUTHOR CONTRIBUTIONS

P.N. designed the algorithm under the supervision of Y.N. P.N. and Y.N. prepared the paper.

COMPETING INTERESTS

The authors declare no competing interests.

ADDITIONAL INFORMATION

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41534-021-00369-3.

Correspondence and requests for materials should be addressed to P.N. or Y.N.

Reprints and permission information is available at http://www.nature.com/reprints

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.